IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No. : 10/814,216 Confirmation No. 6017
Applicant : Randolph L. Campbell
Filed : 3/31/2004
TC/A.U. : 2195
Examiner : Meng Yao Zhe

Docket No. : 42P17827
Customer No. : 8791

Commissioner for Patents
PO Box 1450
Alexandria VA 22313-1450

DECLARATION UNDER 37 C.F.R. § 1.131

Dear Sir:

We, Randolph Campbell and Gehad Galal hereby declare that:

1. I am the inventor of the subject matter claimed in the above-identified patent application, which is assigned to Intel Corporation.

2. This declaration is to establish conception of the invention in the above-identified patent application in the United States, at a date prior to October 1, 2003, the filing date of U.S. Publication No. 2005/0091354 to Lowell, which was cited by the Examiner.

3. We understand that the invention relates to the following:

A. An apparatus comprising:

a processor having a normal execution mode and a host execution mode;

a virtual machine monitor (VMM) implemented in the host execution mode creates original and target protected mode environments to operate guest software in a virtual machine, wherein responsive to a command to switch between the protected modes, the VMM causes the processor to atomically switch between the original protected mode environment and the target protected mode environment; and

a virtual machine control structure (VMCS) to store state information for use in switching between the original protected mode environment and the target protected mode environment, the VMCS to store state information related to the original protected mode environment.

B. A method comprising:

providing a normal execution mode in a processor and a host execution mode in a processor;

creating original and target protected mode environments to operate guest software in a virtual machine utilizing a virtual machine monitor (VMM) implemented in the host execution mode, wherein responsive to a command to switch between the protected modes, atomically switching between the original protected mode environment and the target protected mode environment utilizing the VMM; and

storing state information in a virtual machine control structure (VMCS) for use in switching between the original protected mode environment and the target protected mode environment including storing state information related to the original protected mode environment.

C. A machine-readable medium of a storage device having tangibly stored thereon instructions, which when executed by a machine, cause the machine to perform the following operations comprising:

providing a normal execution mode in a processor and a host execution mode in a processor;

creating original and target protected mode environments to operate guest software in a virtual machine utilizing a virtual machine monitor (VMM) implemented in the host execution mode, wherein responsive to a command to switch between the protected modes, atomically switching between the original protected mode environment and the target protected mode environment utilizing the VMM; and

storing state information in a virtual machine control structure (VMCS) for use in switching between the original protected mode environment and the target protected mode environment including storing state information related to the original protected mode environment.

D. A system comprising:

a processor including virtual machine extension (VMX) instruction support, the processor further having a normal execution mode and a host execution mode;

a virtual machine monitor (VMM) implemented in the host execution mode creates original and target protected mode environments to operate guest software in a virtual machine, wherein responsive to a command to switch between the protected modes, the VMM causes the processor to atomically switch between the original protected mode environment and the target protected mode environment; and

a virtual machine control structure (VMCS) to store state information for use in switching between the original protected mode environment and the target protected mode environment, the VMCS to store state information related to the original protected mode environment.

4. Prior to October 1, 2003, I completed an Invention Disclosure (Exhibit A) describing the invention and submitted the invention disclosure to the legal department of Intel Corporation.

5. After receipt and review of the Invention Disclosure, the legal department of Intel Corporation decided to proceed with the preparation of a patent application and requested that Blakely, Sokoloff, Taylor & Zafman LLP prepare and file a patent application on the subject matter set forth in Exhibit A.

6. Thereafter, the above-identified patent application was prepared with due diligence and filed on March 31, 2004.

We hereby declare that all statements made herein of my own knowledge are true and that the statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Date: 4/21/10

R L Campbell

Randolph L. Campbell

Date: ____________

____

Gehad M. Galal

4. Prior to October 1, 2003, I completed an Invention Disclosure (Exhibit A) describing the invention and submitted the invention disclosure to the legal department of Intel Corporation.

5. After receipt and review of the Invention Disclosure, the legal department of Intel Corporation decided to proceed with the preparation of a patent application and requested that Blakely, Sokoloff, Taylor & Zafman LLP prepare and file a patent application on the subject matter set forth in Exhibit A.

6. Thereafter, the above-identified patent application was prepared with due diligence and filed on March 31, 2004.

We hereby declare that all statements made herein of my own knowledge are true and that the statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Date: ______________ ______________________________
Randolph L. Campbell

Date: 4/21/2010 Gehad M Galal
Gehad M. Galal

EXHIBIT A

[redacted]

INTEL INVENTION DISCLOSURE

ATTORNEY-CLIENT PRIVILEGED COMMUNICATION
located at [redacted]

DATE: [redacted]

[redacted]

SOFTWARE/CTG/MRL

It is important to provide accurate and detailed information on this form. The information will be used to evaluate your invention for possible filing as a patent application. **Invention Disclosure forms MUST be sent electronically via email to your manager/supervisor who should then forward with their approval to our email account "invention disclosure submission."** If you have any questions, please call [redacted]

Last Name: Galal			
Intel Phone Number: [redacted]	First Name: Gehad		M.I. M
E-mail address: [redacted]	Intel Fax Number:		Mailstop: [redacted]
Citizenship: Egyptian	Are you a contractor?	[redacted]	[redacted]
Home Address: 3466 Data Dr. #1223			No: X
City: Rancho Cordova	State: CA	Zip: 95670	Country: US
Corporate Level Group: EPG	Division: CSE		Subdivision: Win NT OS
Supervisor: Randolph Campbell	[redacted]	[redacted]	Phone [redacted]

Last Name: Campbell			
Intel Phone Number: 916-356-4981	First Name: Randolph		M.I. L
E-mail address: [redacted]	Intel Fax Number:		Mailstop: [redacted]
Citizenship: US	Are you a contractor?	[redacted]	[redacted]
Home Address: 904 Halidon Way			No: X
City: Folsom	State: CA	Zip: 95630	Country: US
Corporate Level Group: EPG	Division: CSE		Subdivision: Win NT OS
Supervisor: Andy Vargas	[redacted]	[redacted]	Phone [redacted]

(PROVIDE SAME INFORMATION AS ABOVE FOR EACH ADDITIONAL INVENTOR)

2. Title of Invention:
Context Switch Algorithm Using VMX Features

3. What technology/product/process (code name) does your invention relate to (be specific if you can)
LaGrande Technology, Virtual Machine Extension (VMX), Intel Architecture (IA32) , Prescott, Tejas, Morom, Yonah

4. Include several key words to describe the technology area of the invention in addition to # 3 above:
Virtual Machine, Context Switch, Task Switch

5. Stage of development (i.e. % complete, simulations done, test chips if any, etc.):
Software development complete on simulated machines

[redacted]

If YES, was the manuscript submitted for pre-publication approval through the Author Incentive Program:

If YES, please identify the publication and the date published:

[redacted]

6b. [redacted]s your invention been used/sold or planned to be used/sold by Intel or others? [redacted] product.

[redacted]

6c. Does this invention relate to technology that is or will be covered by a SIG (special interest group)/standard or specification?

[redacted]

6d. If the invention is embodied in a semiconductor device, actual or anticipated date of tapeout?

[redacted]

6e. If the invention is software, actual or anticipated date of any beta tests outside Intel:

[redacted]

7. Was the invention conceived or constructed in collaboration with anyone other than an Intel blue badge employee or in performance of a project involving entities other than Intel (e.g. government, other companies, universities or consortia)? [redacted] [redacted]

8. Is this inventio[redacted] to any other invention disclosure that you have recently submitted? If so, please give the title and inventors: [redacted]

PLEASE READ AND FOLLOW THE DIRECTIONS ON
HOW TO WRITE A DESCRIPTION OF YOUR INVENTION

Try to limit your description to 2-3 pages
Do NOT attach a presentation, white paper, or specification
ANSWER ALL OF THE QUESTIONS BELOW

Please provide a description of the invention and include the following information:

1. Describe in detail what the components of the invention are and how the invention works.

Background:

LaGrande Technology (LT) Virtual Machine Extension (VMX) adds processor support for IA-32 virtual machines on IA-32 processors. Monitor software uses VMX to create one or more IA-32 virtual machines. Guest software (e.g. operating systems, device drivers, applications) may run unmodified inside an IA-32 virtual machine. Certain guest events, instructions and situations trap to the monitor, allowing the monitor to present the guest software with a processor abstraction. A trap from the guest to the monitor is referred to as a VMEXIT. A new instruction, VMCALL, allows the guest software to force a VMEXIT to the monitor. The monitor may resume the guest with the VMRESUME or VMLAUNCH instructions – this transition is referred to as a VMENTER.

The transitions between the monitor and the guest software are controlled by the Virtual Machine Control Structure (VMCS). This structure stores the guest state, the monitor state, and various control registers which determine which guest events trap to the monitor and what state is loaded and stored on VMEXIT and what state is loaded on VMENTER. For example, on VMEXIT guest state is stored to the guest state area of the VMCS and monitor state is loaded from the monitor state area of the VMCS. On VMENTER the guest state is restored from the guest state area in the VMCS. See figure 1. The monitor may read and write fields in the VMCS using the VMREAD and VMWRITE instructions.

VMEXIT and VMENTER transitions switch nearly the entire state of the machine including a new Global Descriptor Table Register (GDTR), a new Interrupt Descriptor Table Register (IDTR), Control registers (CR0, CR3, and CR4), EIP, and ESP.

[redacted]

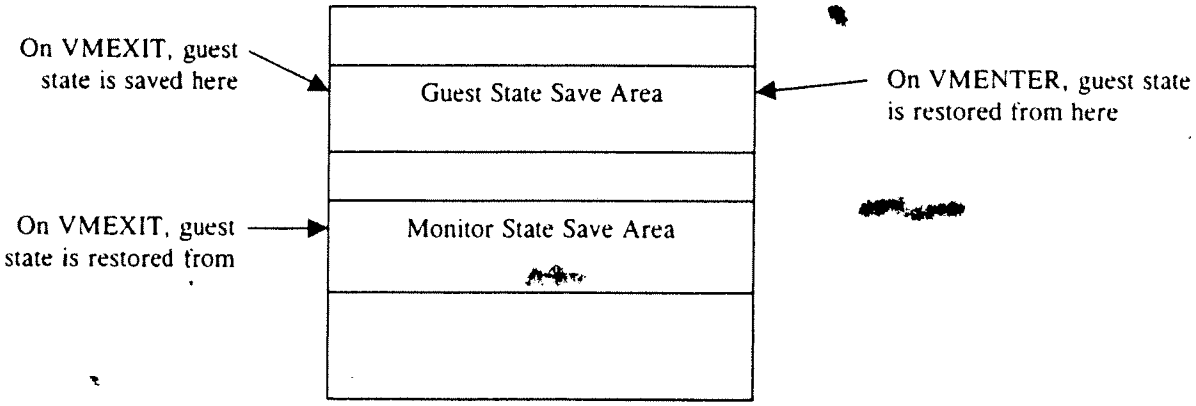
On VMEXIT, guest state is saved here
Guest State Save Area
On VMENTER, guest state is restored from here
On VMEXIT, guest state is restored from
Monitor State Save Area

Figure 1: Virtual Machine Control Structure

Context switching between protected mode environments:

Switching between two protected mode environments can be partially accomplished using the IA-32 hardware task switch mechanism. The IA-32 hardware task mechanism does not update the Global Descriptor Table Register (GDTR) or the Interrupt Descriptor Table Register (IDTR). These registers must be saved and restored by code before or after the task switch. During this code the processor is "in the crack" between the two environments - the processor contains some state from one environment (e.g. the GDTR) and some state from the other environment. Faults or interrupts taken while in the crack may not be handled properly. The following sequence illustrates the problem:

1. Original protected mode environment (O context) is initially operational (with IDTR, GDTR, CR3, and ESP)
2. O context uses a task switch to switch some processor registers to target Protected Mode environment (T context). These registers include CR3, ESP.
3. To use the task switch for loading segment registers, special handling needs to be done to ensure that new segment selector values (loaded by the task switch) are defined properly in the O context GDT.
4. After the task switch is completed, the processor virtual space is changed (by the new CR3), while the processor GDTR and IDTR still contains virtual addresses of the O context.
5. Any interrupt/exception occurring at this point will result in a triple fault (no IDT to handle the interrupt/exception).
6. A lot of complicated processing needs to be done to after the task switch to change the remaining processor registers to the T context.

This invention disclosure presents a mechanism with which VMX features can be used to facilitate the transition between protected mode environments atomically.

Algorithm: Protected Mode Context Switch using VMX Features

The following pseudo code/figure demonstrates the algorithm to perform a complete IA-32 context switch using LT VMX. The transition is made from the Original Protected Mode State (O context) to the Target Protected Mode State (T context). A protected mode state is a set of settings for different processor registers including Control registers (CR0, CR3, CR4), IDTR, GDTR, EIP, and ESP. Note this code assumes the processor is running in protected mode and has VMX enabled (CR4.VMXE=1).

[redacted]

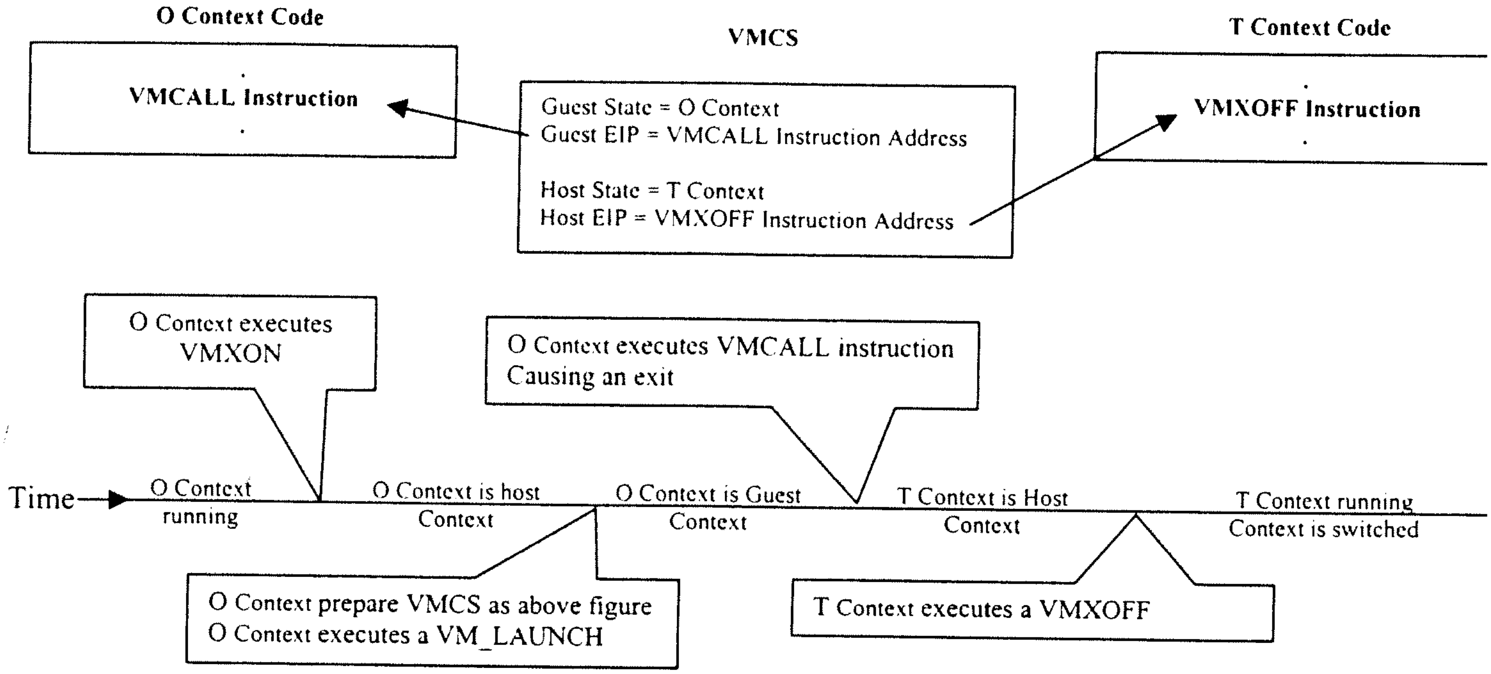
O Context Code
VMCALL Instruction
VMCS
Guest State = O Context
Guest EIP = VMCALL Instruction Address
Host State = T Context
Host EIP = VMXOFF Instruction Address
T Context Code
VMXOFF Instruction
O Context executes VMXON
O Context executes VMCALL instruction Causing an exit
Time
O Context running
O Context is host Context
O Context is Guest Context
T Context is Host Context
T Context running Context is switched
O Context prepare VMCS as above figure
O Context executes a VM_LAUNCH
T Context executes a VMXOFF

Figure 2: VMCS Contents and timeline for context switch

1. O context executes a VMXON instruction. This puts the processor in root/host VMX operation, allowing execution of additional VMX instructions. O context is now running as the VMX host.

2. O context allocates memory for a VMCS, performs a VMCLEAR and a VMPTRLD to initialize and make active the VMCS.

3. O context uses a series of VMWRITE instructions to load the VMCS guest save area with the O context.

4. O Context uses a series of VMWRITE instructions to load the VMCS host save area with the T context. The VMCS guest context area now contains the O context and the host state area contains the T context, see Figure 2.

5. O context loads the guest EIP field of the VMCS guest save area with the address of a VMCALL instruction in the O context (as shown in figure 2).

6. O context loads the EIP field of the VMCS host save area with the address of the entry point in the T context. This first instruction would typically be a VMXOFF instruction.

7. O context executes a VM_LAUNCH instruction. This loads the processor with the VMCS guest state area (which contains the O context).

8. O context now executes in guest context executes a VMCALL instruction forcing a VM-exit.

9. The VM-exit returns the processor to VMX host context, which was prepared to contain the T.

10. While in T context, the processor executes the VMXOFF instruction and continues executing, now running with the desired protected mode state (T context).

Using the above algorithm, the machine is never left in an inconsistent state as with the case in standard task-switch based context switching.

[redacted]

2. **Describe advantage(s) of your invention over what is currently being done.**

Current methods for switching between two different protected mode environments might use the IA-32 hardware task switch, reloading the GDTR or IDTR before or after the task switch. This method puts the processor "in the crack" between two protected mode environments i.e. the processor state is inconsistent for a short period of time. Interrupts happening at this time will not be handled correctly. This algorithm does the transition atomically, thus ensuring a consistent protected mode environment at all times. This improves system reliability and stability.

3. **You MUST include at least one figure illustrating the invention. If the invention relates to software, include a flowchart or pseudo-code representation of the algorithm.**

4. **Value of your invention to Intel (how will it be used?).**

This algorithm may be used anytime a programmer wishes to transition from one protected mode environment to another. An example of that is starting switching back from Virtual Machine Monitor to one of its guest VMs after teardown.

5. **Explain how your invention is novel. If the technology itself is not new, explain what makes it different.**

This algorithm performs an atomic mega-task switch using LT VMX. Older methods perform the machine state transition in several steps. The older method requires mapping the GDT and IDT in both environment at any point in time while switching between them, this may not be possible if the two environments do not have the same virtual address regions free.

6. **Identify the closest or most pertinent prior art that you are aware of.**

7. **Who is likely to want to use this invention or infringe the patent if one is obtained and how would infringement be detected?**

[redacted]

D